In the Claims:

Please amend claims 1-11, 13-24, 26-27, 29-33, and 35- 48 as indicated below.

1. (Currently amended) A method for handling events in a distributed computing environment, comprising:

receiving, by an event message endpoint on a client platform in the distributed computing environment, indications from one or more client processes registering interest in receiving one or more of a plurality of events generated by a service in the distributed computing environment;

the event message endpoint automatically subscribing to the one or more events with the service in response to said indications registering interest in the one or more events received from the one or more client processes;

receiving, by the event message endpoint, a message in a ~~data representation~~ markup language sent to [[a]] the client platform in the distributed computing environment from [[a]] the service in the distributed computing environment, wherein the message includes a ~~data representation~~ markup language representation of ~~an event~~ one of the one or more events generated by the service to which the event message endpoint is subscribed; and

sending, by the event message endpoint, the ~~data representation~~ markup language representation of the event to at least one of the one or more client processes registered with the event message endpoint to receive the event ~~from the service~~.

2. (Currently amended) The method as recited in claim 1, further comprising, prior to said receiving, by an event message endpoint on a client platform in the

distributed computing environment, indications from one or more client processes on the client platform registering interest in receiving one or more of a plurality of events generated by a service in the distributed computing environment:

> ~~receiving~~ obtaining a ~~data representation~~ markup language schema on the client platform, wherein said ~~data representation~~ markup language schema defines a message interface for ~~a set~~ the plurality of events generated by the service; and

> ~~generating~~ automatically constructing [[an]] the event message endpoint for the client platform according to the ~~data representation~~ markup language schema, ~~wherein said receiving a message and said sending the data representation language representation of the event to one or more processes are performed by the event message endpoint~~ wherein said constructing is performed within a runtime environment of the client platform.

3. (Currently amended)   The method as recited in claim [[2]] 1, further comprising ~~the event message endpoint subscribing to one or more of the set of events generated by the service, wherein~~ the service ~~is configured to send~~ sending one or more messages each including ~~data representation~~ a markup language representation[[s]] of an event to subscribers to the event ~~when~~ in response to generation of the event ~~is generated~~ by the service.

4. (Currently amended)   The method as recited in claim [[2]] 1, wherein the ~~data representation~~ markup language message from the service includes an authentication credential for the service, the method further comprising the event message endpoint authenticating the markup language message as being from the service according to ~~using~~ the authentication credential for the service ~~to authenticate the data representation language message as being from the service~~.

5. (Currently amended)   The method as recited in claim [[2]] 1, further comprising the event message endpoint verifying type correctness of the ~~data representation~~ markup language message according to the ~~data representation~~ markup language schema ~~subsequent to said receiving a message~~ prior to said sending the markup language representation of the event to the at least one of the one or more client processes.

6. (Currently amended)   The method as recited in claim [[2]] 1, wherein the ~~data representation~~ markup language schema defines ~~a set of messages that the service may send to the event message endpoint~~ a plurality of messages including markup language representations of the plurality of events generated by the service, the method further comprising the event message endpoint verifying ~~the~~ correctness of the ~~data representation~~ markup language message from the service according to the ~~data representation~~ markup language schema prior to said sending the markup language representation of the event to the at least one of the one or more client processes.

7. (Currently amended)   The method as recited in claim 2, ~~further comprising each of the one or more processes registering interest in one or more of the set of events generated by the service with the event message endpoint subsequent to said generating an event message endpoint~~ wherein said constructing said event message endpoint is performed by computer-executable message endpoint construction code on the client platform.

8. (Currently amended)   The method as recited in claim [[7]] 1,

wherein said receiving indications from one or more client processes registering interest in receiving one or more of the plurality of events comprises receiving from each of the one or more client processes ~~providing~~ an event handler callback method for an event handler of the respective client process ~~to the event message endpoint~~; and

wherein said sending the ~~data representation~~ <u>markup</u> language representation of the event to <u>at least one of the</u> one or more <u>client</u> processes registered <u>with the event message endpoint</u> to receive the event ~~from the service~~ comprises:

the event message endpoint calling an event handler <u>callback</u> method of each <u>client</u> process registered with the event message endpoint to <u>receive</u> the event; and

the event message endpoint passing the ~~data representation~~ <u>markup</u> language representation of the event to each called event handler.

9. (Currently amended) The method as recited in claim [[7]] <u>1</u>, further comprising:

[[a]] <u>at least one client</u> process unregistering interest in a first event of the service <u>with the event message endpoint</u>; and

the event message [[gate]] <u>endpoint automatically</u> unsubscribing to the first event with the service ~~subsequent~~ <u>in response</u> to said unregistering <u>interest by the at least one client process</u>;

wherein the service is ~~further~~ configured to not send messages including ~~data representation~~ <u>markup</u> language representations of the first event to event message endpoints that are unsubscribed to the first event.

10. (Currently amended) The method as recited in claim 2, ~~further comprising~~ <u>wherein said obtaining a markup language schema comprises</u> receiving the ~~data representation~~ <u>markup</u> language schema of the service in a service advertisement of the service, <u>wherein the service advertisement is a markup language document that defines each event message generated by the service</u>.

11. (Currently amended)   The method as recited in claim 1, wherein the one or more <u>client</u> processes are executing within the client platform.

12. (Original) The method as recited in claim 1, wherein the event is a Java event.

13. (Currently amended)   The method as recited in claim 1, wherein said ~~data representation~~ <u>markup</u> language is eXtensible Markup Language (XML).

14. (Currently amended) A device, comprising:

a processor; <u>and</u>

a memory coupled to said processor[[;]]<u>, wherein said memory comprises program instructions executable by said processor to implement</u> an event message gate unit configured to:

<u>receive indications from one or more client processes registering interest in receiving one or more of a plurality of events generated by a service in a distributed computing environment;</u>

<u>automatically subscribe to the one or more events with the service in response to said indications registering interest in the one or more events received from the one or more client processes;</u>

receive a message in a ~~data representation~~ <u>markup</u> language sent to the device in the distributed computing environment from [[a]] <u>the</u> service in the distributed computing environment, wherein the message includes a ~~data representation~~ <u>markup</u> language

representation of ~~an event~~ one of the one or more events generated by the service to which the event message gate unit is subscribed; and

send the ~~data representation~~ markup language representation of the event to at least one of the one or more client processes registered with the event message gate unit to receive the event ~~from the service~~.

15. (Currently amended)   The device as recited in claim 14, wherein, to implement the event message gate unit on the device, the program instructions are executable by said processor to:

~~receive~~ obtain a ~~data representation~~ markup language schema, wherein said ~~data representation~~ markup language schema defines a message interface for ~~a set~~ the plurality of events generated by the service; and

~~generate~~ automatically construct the event message gate unit according to the ~~data representation~~ markup language schema within a runtime environment on the device.

16. (Currently amended)   The device as recited in claim [[15]] 14, wherein the event message gate unit is further configured to verify type correctness of the ~~data representation~~ received markup language message according to the ~~data representation~~ markup language schema ~~subsequent to said receiving a message~~ prior to said sending the markup language representation of the event to the at least one of the one or more client processes.

17. (Currently amended)   The device as recited in claim [[15]] 14, wherein the ~~data representation~~ markup language schema defines ~~a set of messages that the service may send to the event message gate unit~~ a plurality of messages including markup language representations of the plurality of events generated by the service, and wherein

the event message gate unit is further configured to verify ~~the~~ correctness of the ~~data representation~~ <u>received markup</u> language message ~~from the service~~ according to the ~~data representation~~ <u>markup</u> language schema <u>prior to said sending the markup language representation of the event to the at least one of the one or more client processes</u>.

18. (Currently amended) The device as recited in claim 15, wherein<u>, to obtain a markup language schema,</u> the device is ~~further~~ configured to receive the ~~data representation~~ <u>markup</u> language schema of the service in a service advertisement of the service<u>, wherein the service advertisement is a markup language document that defines each event message generated by the service</u>.

19. (Currently amended) The device as recited in claim 14, ~~wherein the event message gate unit is further configured to subscribe to one or more of the set of events generated by the service, and~~ wherein the service is configured to send <u>one or more</u> messages <u>each</u> including ~~data representation~~ <u>a markup</u> language representation~~[[s]]~~ of an event to subscribers to the event ~~when~~ <u>in response to generation of</u> the event ~~is generated~~ <u>by the service</u>.

20. (Currently amended) The device as recited in claim 14, wherein the ~~data representation~~ <u>markup</u> language message from the service includes an authentication credential for the service, wherein the event message gate unit is further configured to <u>authenticate the markup language message as being from the service</u> ~~use~~ <u>according to</u> the authentication credential for the service ~~to authenticate the data representation language message as being from the service~~.

21. (Currently amended) The device as recited in claim [[14]] <u>15</u>, ~~wherein each of the one or more processes are configured to register interest in one or more of the set of events generated by the service with the event message gate unit subsequent to said generating an event message gate unit~~ <u>wherein, to implement the event message gate unit on the device, the program instructions are further executable by said processor to</u>:

obtain an address for said service within the distributed computing environment;

obtain an authentication credential indicating authorization to access said service; and

construct the event message gate unit according to the markup language schema, the obtained address for the service, and the obtained authentication credential for the service.

22. (Currently amended)   The device as recited in claim [[21]] 14,

wherein, to receive indications from one or more client processes in said registering interest in receiving one or more of the set plurality of events, the event message gate unit is configured to receive from each of the one or more client processes is configured to provide an event handler callback method for an event handler of the respective client process to the event message gate unit; and

wherein, in said sending the data representation to send the markup language representation of the event to at least one of the one or more client processes registered with the event message gate unit to receive the event from the service, the event message gate unit is further configured to:

call an event handler callback method of each client process registered with the event message gate unit to receive the event; and

pass the data representation markup language representation of the event to each called event handler.

23. (Currently amended)   The device as recited in claim [[21]] 14,

~~wherein a first process is configured to unregister interest in a first event of the service;~~

wherein the event message gate unit is further configured to:

receive an indication from at least one client process unregistering interest in a first event of the service with the event message gate unit; and

automatically unsubscribe to the first event with the service ~~subsequent~~ in response to said indication from the at least one client process unregistering interest in the first event; and

wherein the service is configured to not send messages including ~~data representation~~ markup language representations of the first event to event message gate units that are unsubscribed to the first event.

24. (Currently amended)   The device as recited in claim 14, wherein the one or more client processes are executing ~~within the client platform~~ on the device.

25. (Original) The device as recited in claim 14, wherein the event is a Java event.

26. (Currently amended)   The device as recited in claim 14, wherein said ~~data representation~~ markup language is eXtensible Markup Language (XML).

27. (Currently amended) A device, comprising:

a processor;

a memory coupled to said processor[[;]], wherein said memory comprises program instructions executable by said processor to implement a service process configured to:

generate an event;

generate a message in a ~~data representation~~ markup language, wherein the message includes a ~~data representation~~ markup language representation of the event generated by the service process; and

send the message to one or more event message gate units in [[the]] a distributed computing environment that have each automatically subscribed to the event with the service process in response to one or more client processes registering interest in the event with the respective event message gate unit;

wherein each of the one or more event message gate units are operable to distribute the ~~data representation~~ markup language representation of the event sent in the message from the service process to the one or more client processes registered with the respective event message gate unit to receive the event from the service process.

28. (Original) The device as recited in claim 27, wherein the device further comprises a service message gate unit, wherein said generating a message and said sending the message are performed by the service message gate unit on behalf of the service process.

29. (Currently amended)   The device as recited in claim 27, wherein the service process is further configured to:

provide a ~~data representation~~ <u>markup</u> language schema <u>within the distributed</u> <u>computing environment</u>, wherein said ~~data representation~~ <u>markup</u> language schema defines a message interface for a ~~set~~ <u>plurality</u> of events generated by the service <u>process</u>; and

wherein <u>each of</u> the one or more event message gate units ~~are generated~~ <u>is</u> <u>constructed</u> according to the ~~data representation~~ <u>markup</u> language schema <u>within a runtime environment on a respective device within the distributed</u> <u>computing environment prior to said one or more client processes</u> <u>registering interest in the event with the respective event message gate</u> <u>unit</u>.

30. (Currently amended) The device as recited in claim 29, wherein the ~~data~~ ~~representation~~ <u>markup</u> language schema defines ~~a set of messages that the service may~~ ~~send to the event message gate units~~ <u>a plurality of messages including markup language</u> <u>representations of the plurality of events generated by the service process</u>.

31. (Currently amended) The device as recited in claim 29, wherein the service process is further configured to provide the ~~data representation~~ <u>markup</u> language schema in a service advertisement<u>, wherein the service advertisement is a markup</u> <u>language document that defines each event message generated by the service process</u>.

32. (Currently amended) The device as recited in claim 27, wherein the service process is further configured to send <u>one or more</u> messages <u>each</u> including ~~data~~ ~~representation~~ <u>a markup</u> language representation[[s]] of an event to event message gate units subscribed to the event ~~when~~ <u>in response to generation of</u> the event ~~is generated~~ by the service process.

33. (Currently amended) The device as recited in claim 27, wherein the service process is further configured to attach an authentication credential for the service <u>process</u> to the ~~data representation~~ <u>markup</u> language message, wherein the <u>event message</u>

gate units that receive the markup language message are configured to ~~authentication credential is configured for use in authenticating the data representation~~ authenticate the markup language message as being from the service process according to the authentication credential attached to the message.

34. (Original) The device as recited in claim 27, wherein the events are Java events.

35. (Currently amended)   The device as recited in claim 27, wherein said ~~data representation~~ markup language is eXtensible Markup Language (XML).

36. (Currently amended)   A ~~tangible~~ non-transitory computer readable medium comprising program instructions, wherein the program instructions are computer-executable to implement an event message endpoint on a client platform in a distributed computing environment, wherein the event message endpoint is configured to:

receive indications from one or more client processes registering interest in receiving one or more of a plurality of events generated by a service in the distributed computing environment;

automatically subscribe to the one or more events with the service in response to said indications registering interest in the one or more events received from the one or more client processes;

~~receiving~~ receive a message in a ~~data representation~~ markup language sent to [[a]] the client platform in the distributed computing environment from [[a]] the service in the distributed computing environment, wherein the message includes a ~~data representation~~ markup language representation of ~~an event~~ one of the one or more events generated by the service to which the event message endpoint is subscribed; and

~~sending~~ <u>send</u> the ~~data representation~~ <u>markup</u> language representation of the event to <u>at least one of the</u> one or more <u>client</u> processes registered <u>with the event message endpoint</u> to receive the event ~~from the service~~.

37. (Currently amended)   The ~~tangible~~ <u>non-transitory</u> computer readable medium as recited in claim 36, wherein<u>, to implement the event message endpoint on the client platform,</u> the program instructions are ~~further~~ computer-executable to ~~implement~~:

~~receiving~~ <u>obtain</u> a ~~data representation~~ <u>markup</u> language schema ~~on the client platform~~, wherein said ~~data representation~~ <u>markup</u> language schema defines a message interface for ~~a set~~ <u>the plurality</u> of events generated by the service; and

~~generating~~ <u>automatically construct</u> [[an]] <u>the</u> event message endpoint for the client platform according to the ~~data representation~~ <u>markup</u> language schema, ~~wherein said receiving a message and said sending the data representation language representation of the event to one or more processes are performed by the event message endpoint~~ <u>wherein said constructing is performed within a runtime environment of the client platform</u>.

38. (Currently amended)   The ~~tangible~~ <u>non-transitory</u> computer readable medium as recited in claim [[37]] <u>36</u>, ~~wherein the program instructions are further computer-executable to implement the event message endpoint subscribing to one or more of the set of events generated by the service,~~ wherein the service is configured to send <u>one or more</u> messages <u>each</u> including ~~data representation~~ <u>a markup</u> language representation[[s]] of an event to subscribers to the event ~~when~~ <u>in response to generation of</u> the event ~~is generated~~ <u>by the service</u>.

39. (Currently amended)   The ~~tangible~~ <u>non-transitory</u> computer readable medium as recited in claim [[37]] <u>36</u>, wherein the ~~data representation~~ <u>markup</u> language

message from the service includes an authentication credential for the service, wherein ~~the program instructions are further computer-executable to implement~~ the event message endpoint <u>is configured to authenticate the markup language message as being from the service according to</u> ~~using~~ the authentication credential for the service ~~to authenticate the data representation language message as being from the service~~.

40. (Currently amended)   The ~~tangible~~ <u>non-transitory</u> computer readable medium as recited in claim [[37]] <u>36,</u> wherein ~~the program instructions are further computer-executable to implement~~ the event message endpoint <u>is configured to verify</u> ~~verifying~~ type correctness of the ~~data representation~~ <u>markup</u> language message according to the ~~data representation~~ <u>markup</u> language schema ~~subsequent to said receiving a message~~ <u>prior to said sending the markup language representation of the event to the at least one of the one or more client processes</u>.

41. (Currently amended)   The ~~tangible~~ <u>non-transitory</u> computer readable medium as recited in claim [[37]] <u>36,</u> wherein the ~~data representation~~ <u>markup</u> language schema defines ~~a set of messages that the service may send to the event message endpoint~~ <u>a plurality of messages including markup language representations of the plurality of events generated by the service</u>, wherein ~~the program instructions are further computer-executable to implement~~ the event message endpoint <u>is configured to verify</u> ~~verifying the~~ correctness of the ~~data representation~~ <u>markup</u> language message from the service according to the ~~data representation~~ <u>markup</u> language schema <u>prior to said sending the markup language representation of the event to the at least one of the one or more client processes</u>.

42. (Currently amended)   The ~~tangible~~ <u>non-transitory</u> computer readable medium as recited in claim 37, wherein,<u> to implement the event message endpoint on the client platform,</u> the program instructions are further computer-executable to<u>:</u> ~~implement each of the one or more processes registering interest in one or more of the set of events generated by the service with the event message endpoint subsequent to said generating an event message endpoint~~

obtain an address for said service within the distributed computing environment;

obtain an authentication credential indicating authorization to access said service; and

construct the event message endpoint according to the markup language schema, the obtained address for the service, and the obtained authentication credential for the service.

43. (Currently amended)   The ~~tangible~~ non-transitory computer readable medium as recited in claim [[42]] 36,

wherein, to receive indications from one or more client processes ~~in said~~ registering interest in receiving one or more of the ~~set~~ plurality of events, ~~the program instructions are further computer-executable to implement~~ the event message endpoint is configured to receive from each of the one or more client processes ~~providing~~ an event handler callback method for an event handler of the respective client process ~~to the event message endpoint~~;

wherein, ~~in said sending the data representation~~ to send the markup language representation of the event to at least one of the one or more client processes registered with the event message endpoint to receive the event ~~from the service~~, the ~~program instructions are further computer-executable to implement~~ event message endpoint is configured to:

~~the event message endpoint calling~~ call an event handler callback method of each client process registered with the event message endpoint to receive the event; and

~~the event message endpoint passing~~ pass the ~~data representation~~ markup language representation of the event to each called event handler.

44. (Currently amended) The ~~tangible~~ non-transitory computer readable medium as recited in claim [[42]] 36, ~~wherein the program instructions are further computer-executable to implement:~~

~~a process unregistering interest in a first event of the service; and~~

wherein the event message endpoint is further configured to:

receive an indication from at least one client process unregistering interest in a first event of the service with the event message endpoint; and

automatically unsubscribe ~~the event message gate unsubscribing~~ to the first event with the service ~~subsequent~~ in response to said indication from the at least one client process unregistering interest in the first event;

wherein the service is ~~further~~ configured to not send messages including ~~data representation~~ markup language representations of the first event to event message endpoints that are unsubscribed to the first event.

45. (Currently amended) The ~~tangible~~ non-transitory computer readable medium as recited in claim 37, wherein, to obtain a markup language schema, the program instructions are further computer-executable to ~~implement receiving~~ receive the ~~data representation~~ markup language schema of the service in a service advertisement of the service, wherein the service advertisement is a markup language document that defines each event message generated by the service.

46. (Currently amended) The ~~tangible~~ non-transitory computer readable

medium as recited in claim 36, wherein the one or more <u>client</u> processes are executing within the client platform.

47. (Currently amended)   The ~~tangible~~ <u>non-transitory</u> computer readable medium as recited in claim 36, wherein the event is a Java event.

48. (Currently amended)   The ~~tangible~~ <u>non-transitory</u> computer readable medium as recited in claim 36, wherein said ~~data representation~~ <u>markup</u> language is eXtensible Markup Language (XML).